# Подреждане на функции в програма

Програма, която въвежда програма на C++, разпознава всички дефиниции на функции (процедури) и обръщенията към тях и прави опит за установяване на топологична наредба на функциите.След като установи наредбата, програмата да прави пренареждане на функциите въз основа на топологичното сортиране.
В случай, че наредба не е възможна да се установи поради наличие на циклични зависимости, програмата да предлага вмъкването на минимален брой декларации така, че програмата да се компилира успешно.
За опростяване считаме, че:

- програмата се състои само от един .cpp файл;
- в програмата няма класове;
- в програмата няма глобални променливи и константи

**class Function –** описва дадена функция **:**

```
private:
string name – име на функцията
string declaration – декларация на функцията
string contents – съдържание на функцията
int start,end- номера на начален и краен ред на функцията в подадения файл

void readFunction(string) – прочита функция от даден стринг
public:

Function(string);

Function(vector<string>, int, int);

const string& getName() const – връща името на функцията

const string& getContents() const – връща съдържанието на функцията

const string& getDeclaration() const – връща декларацията на функцията

int getEnd() const – връща номера на началния ред в подадения файл

int getStart() const – връща номера на крайния ред в подадения файл
```

**class Graph -** представя граф**:**

**private:**
map<T, int> mapping – асоциира всеки обект с индекс в матрицата
map<int, T> reverse_mapping – обратното на mapping
vector<vector<bool> > matrix – матрица на съседство

void resize(int size) – преоразмерява матрицата

**public:**
**Graph()** - създава празен граф

**Graph(const vector<T>& v) –** създава граф с върхове елементите на v

vector<int> **neighbours**(int v) – връща всички съседи на върха v

bool **has_cycle**(int v) – проверява дали върха v участва в цикъл

bool **has_incoming_edges**(int v) – проверява дали във върха v завършват ребра

vector<T> **remove_cycles**() - премахва върховете, които участват в цикли и ги връща. Деструктивна операция!!

void **add_edge**(T f, T s) – добавя ребро между f и s

void **remove_edge**(T f, T s) – премахва реброто между f и s ако съществува

vector<T> **topologicalSort**() - връща върховете, сортирани със топологическо сортиране

void **print**() - принтира графа

**class FunctionOrder –** използва се за пренареждане на функциите**:**
map<string, func_ptr> functions – държи имена на функции и техните указатели
int b – на кой ред започва първата функция с дадения файл
vector<string> file – съдържа целия файл
Graph<func_ptr> g – граф, който се използва за топологическо сортиране на функциите
string filename – име на файла

pair<int, int> **getFunctionText**(vector<string> text, int s, int e) – връща начален и краен ред на първата срещната функция в интервала [s,e]

void **loadFile**(string) – зарежда файла от диска

vector<func_ptr> **findFunctionCalls**(func_ptr f) – връща указатели към всички функции които се извикват от дадената
public:

**FunctionOrder**(string filename) – конструктор с параметър име на файл

void **orderFunctions**() - пренарежда функциите и ги записва във файл с име new.filename, където filename е името на файла